

I. REAL PARTY IN INTEREST

The subject application is owned by Advanced Micro Devices Corporation, a corporation organized and existing under and by virtue of the laws of the State of Delaware, and having its principal place of business at One AMD Place, Sunnyvale, CA 94088, as evidenced by the assignment recorded at Reel 014593, Frame 0736.

II. RELATED APPEALS AND INTERFERENCES

No other appeals or interferences are known which would directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

Claims 1-2, 4-11, 13-20, 22-31 and 96 are pending in the present application. Claims 1-2, 4-11, 13-20, 22-31 and 96 stand finally rejected and are the subject of this appeal. A clean copy of claims 1-2, 4-11, 13-20, 22-31 and 96, as on appeal (incorporating all amendments), is included in the Appendix hereto.

Claims 3, 12, 21, and 32-95 were cancelled during the prosecution of this application, and thus are not a subject of this appeal.

IV. STATUS OF AMENDMENTS

No amendment to the claims has been filed subsequent to the final rejection. The Appendix hereto reflects the current state of the rejected claims.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

Independent claim 1 is directed to an SMBus host controller (e.g. Fig. 23, 200; specification, page 12, lines 10-19) comprising an SMBus interface (e.g., Fig. 23, 213, 214; page 12, lines 18-19) and an SMBus message handler (e.g., Fig. 23, 202, SMB_PRTCL, 203, 201, 207; page 12, lines 10-19). The SMBus message handler includes a memory (e.g., Fig. 23, 202; page 12, lines 14-15) storing microcode comprising at least two programs each for handling a bus command protocol, each program comprising at least one instruction. The SMBus message handler further includes an interface to a register (e.g., Fig. 23, SMB_PRTCL and interface thereto from 207; page 12, lines 20-29) configured to identify a starting address of a program in said memory. An instruction fetch unit (e.g.; Fig. 23, 203; page 14, lines 16-18) of the SMBus message handler is configured to read an instruction at an address in said memory, said address being specified by a program counter (e.g., Fig. 23, 203; page 12, lines 30-32). A finite-state machine (e.g.; Fig. 23, 201; page 12, lines 1-5) of the SMBus message handler is configured to receive and interpret the instructions read by said instruction fetch unit and manage the data transfer between the SMBus interface and a register set in compliance with said instructions read from said memory. The SMBus message handler also includes an address register array (e.g., Fig. 23, 207; page 12, lines 26-29) comprising a plurality of starting addresses of programs stored in said memory, said register comprising an offset for pointing at a specific register in said address register array.

Independent claim 10 is directed to an integrated circuit chip (e.g. Fig. 23, 200; specification, page 12, lines 10-19) for transmitting and receiving data over an SMBus, including an SMBus interface (e.g., Fig. 23, 213, 214; page 12, lines 18-19), an interface to a memory (e.g., Fig. 23, 202 and interfaces thereto from 207, 203, and 201; page 12, lines 14-15) storing microcode comprising at least two programs each for handling a bus command protocol, each program comprising at least one instruction. The integrated circuit chip further includes an interface to a register (e.g., Fig. 23, SMB_PRTCL and interface thereto from 207; page 12, lines 20-29) configured to identify a starting address

of a program in said memory. An instruction fetch unit (e.g.; Fig. 23, 203; page 14, lines 16-18) of the integrated circuit chip is configured to read an instruction at an address in said memory; said address being specified by a program counter (e.g., Fig. 23, 203; page 12, lines 30-32). A finite-state machine (e.g.; Fig. 23, 201; page 12, lines 1-5) of the integrated circuit chip is configured to receive and interpret the instructions read by said instruction fetch unit and to manage the data transfer between the SMBus interface, and a register set in compliance with said instructions read from said memory. An address register array (e.g., Fig. 23, 207; page 12, lines 26-29) of the integrated circuit chip comprises a plurality of starting addresses of programs stored in said memory, said register comprising an offset for pointing at a specific register in said address register array.

Independent claim 19 is directed to a method for controlling an SMBus (e.g., Fig. 26; page 16, lines 11-12) comprising storing, in a memory, microcode (e.g., Fig. 23, 202; page 12, lines 14-15) comprising at least two programs each for handling a bus command protocol, each program comprising at least one instruction and identifying a starting address (e.g., Fig. 26, 402; page 16, lines 12-15) of one of the at least two programs stored in the-memory. The method further comprises fetching instructions (e.g., Fig. 26, 402; page 16, lines 12-15 and page 17, lines 4-5) of said program one after another and receiving the instructions into a finite-state machine. The method further includes interpreting the instructions (e.g., Fig. 26, 403, 404, 405; page 16, lines 16-21). The method further includes transferring data between an SMBus interface and a register set in compliance with the instruction present in said finite-state machine (e.g., Fig. 26, 408, 409 and 420 421; page 17, lines 24-26 and page 18, lines 8-11). The method further includes reading a first value of a protocol register specifying an offset in an address register array (e.g., Fig. 26, 405, 406 407; page 16, lines 17-23 and page 14 lines 24-27). The method further comprises reading a second value of a register of said address register array, said register being specified by said offset, the second value constituting said starting address of said program (e.g., Fig. 26, 405, 406, 407; page 16, lines 17-23 and page 14 lines 24-27).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Claims 1, 6-10, 15-19, 28-31 and 96 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Stancil, U.S. Patent 7,149,927, in view of Luke, U.S. Patent 6,505,267 and in further view of Steely, U.S. Patent 5,581,719.

2. Claims 2, 4, 5, 11, 13, 14, 20, 22-27 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Stancil in view of Luke and in further view of Applicant's Admitted Prior Art (AAPA).

VII. ARGUMENT

A. Claims 1, 6-9 and 96

The Examiner rejected claims 1, 6-9, and 96 as being unpatentable over Stancil, U.S. Patent 7,149,927, in view of Luke, U.S. Patent 6,505,267 and in further view of Steely, U.S. Patent 5,581,719. Appellant respectfully submits that the Examiner's rejection is erroneous for at least the following reasons.

Appellant respectfully submits that Stancil in view of Luke and in further view of Steely fails to teach or suggest "a finite-state machine configured to receive and interpret the instructions read by said instruction fetch unit and manage the data transfer between the SMBus interface and a register set in compliance with said instructions read from said memory" as recited in independent claim 1. In the Advisory Action of June 2, 2009, the Examiner contends the following:

Stancil's state machine alone would not "interpret" the instructions read from the instruction fetch unit in part because Stancil does not teach IFU. It is the position of the Examiner that the State machine taught by Stancil functions as a logic that would be inherently capable of reading and processing (interpret) instructions read through the instruction fetch unit as combined by the secondary references of Luke and Steele. The Op code of Luke functions as an instruction fetch unit as it stores the current

instruction for the sequencer commands that are addressed by the program counter. Further, the sequencer coupled with the op code (Col. 9, Lines 14-20) retrieve instructions from the external memory. Therefore it is the position of the Examiner that collectively the op-code and sequencer 46 perform the functions that of an instruction fetch unit as addressed in the claimed limitation. (Emphasis added).

Appellant agrees with the Examiner's statement that Stancil's state machine alone would not interpret instructions read from an instruction fetch unit, and that Stancil does not teach an instruction fetch unit. However, Appellant respectfully disagrees with the Examiner's statement that the "[s]tate machine taught by Stancil ... would be inherently capable of reading and processing (interpret) instructions." Appellant respectfully submits that the Examiner has provided no basis to support such a contention, in neither the advisory action nor the final office action of May 5, 2009. With regard to inherency, MPEP 2112 states, in pertinent part:

"In relying upon the theory of inherency, the examiner must provide a basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristic necessarily flows from the teachings of the applied prior art." *Ex parte Levy*, 17 USPQ2d 1461, 1464 (Bd. Pat. App. & Inter. 1990) (emphasis in original)

Despite contending that the state machine of Stancil is "inherently capable of reading and processing (interpret) instructions," the Examiner has provided no basis in fact and/or technical reasoning as to why this allegedly inherent capability necessarily flows from the applied prior art, per the requirements of MPEP 2112 noted above. With respect to elements 112 and 116 of Stancil (which the Examiner reads on Applicant's recited "finite-state machine"), Stancil states the following in col. 4, lines 14-32:

FIGS. 2-5 provide further detail regarding SMBus-to-JTAG emulator 110 pertaining to how the emulator converts between SMBus and JTAG. A more detailed block diagram of emulator 110 is shown in FIG. 2. As shown therein, the emulator 110 preferably comprises an SMBUs state machine 112, an SMBus packet decoder/encoder 114 and a JTAG

interface logic state machine 116. The SMBus state machine 112 comprises logic that receives SMBus packets from the host test system 102 and, with the help of the SMBus packet decoder/encoder 114, extracts the information from the packets necessary for the JTAG logic 122. The decoder/encoder 114 then creates JTAG-compliant communications that are sent to the JTAG logic 122 associated with the DUT 120 under the control of the JTAG interface logic state machine 116. Similarly, JTAG communications from the JTAG logic 122 are received by state machine 116, decoded by decoder/encoder 114 and are converted to SMBus-compliant packets by decoder/encoder 114 and provided to the host test system 102 under the control of the SMBus state machine 112. (Emphasis added).

Nothing in the above citation teaches, much less suggests, that either of state machines 112 or 116 is “inherently capable of reading and processing [interpreting] instructions.” Absent any teaching or suggestion in the above quotation, and absent any technical reasoning as to why the recited combination of features is inherent, Appellant submits that the alleged inherency has not been established. Appellant further submits that the Examiner is relying on speculation regarding the inherency of recited features discussed above, and notes that mere speculation is not sufficient to support a *prima facie* case of obviousness. *See In re Warner*, 379 F.2d 1011, 1017, 154 USPQ 173, 178 (CCPA 1967); *In re Sporck*, 301 F.2d 686, 690, 133 USPQ 360, 364 (CCPA 1962).

Appellant further notes that other than the quotation above, Stancil provides no additional discussion regarding state machines 112 and 116. Thus, the Stancil does not provide any teaching or suggestion that state machines 112 and/or 116 “manage the data transfer ... in compliance with said instructions read from said memory” as recited in independent claim 1. Furthermore, neither Luke nor Steely provides any teaching or suggestion that, taken singly or in combination with the other references, results in this combination of features. Thus, lacking any additional teachings regarding state machines 112 and/or 116, as well as any technical reasoning as to why either or both of these state machines are “inherently capable of reading and processing [interpreting] instructions,” much less any teaching that state machines 112 and/or 116 are “configured to ... manage the data transfer between the SMBus interface and a register set in compliance with said

instructions read from said memory,” it follows that Stancil in view of Luke and Steely fail to teach or suggest all of the elements of the independent claims, including “a finite-state machine configured to receive and interpret the instructions read by said instruction fetch unit and manage the data transfer between the SMBus interface and a register set in compliance with said instructions read from said memory.”

For at least the reasons given above, Appellant respectfully submits that the rejection of claim 1 is erroneous. Accordingly, Appellant respectfully requests reversal of the § 103 rejection of claim 1 and its associated dependent claims 6-9 and 96.

B. Claims 10 and 15-18

The Examiner rejected claims 10 and 15-18 as being unpatentable over Stancil, U.S. Patent 7,149,927, in view of Luke, U.S. Patent 6,505,267 and in further view of Steely, U.S. Patent 5,581,719. Appellant respectfully submits that the Examiner’s rejection is erroneous for at least the following reasons.

Independent claim 10 recites a combination of features including elements similar to those argued above with respect to claim 1. Claims 15-18 depend from claim 10, and thus incorporate its features. Accordingly, for at least the same reasons as given above with respect to claim 1, Appellant respectfully requests reversal of the § 103 rejection of claims 10 and 15-18.

C. Claims 19 and 28-31

The Examiner rejected claims 19 and 28-31 as being unpatentable over Stancil, U.S. Patent 7,149,927, in view of Luke, U.S. Patent 6,505,267 and in further view of Steely, U.S. Patent 5,581,719. Appellant respectfully submits that the Examiner’s rejection is erroneous for at least the following reasons.

Independent claim 19 recites a combination of features including elements similar to those argued above with respect to claim 1. Claims 28-31 depend from claim 19, and

thus incorporate its features. Accordingly, for at least the same reasons as given above with respect to claim 1, Appellant respectfully requests reversal of the § 103 rejection of claims 19 and 28-31.

D. Claims 2, 4, and 5

The Examiner rejected claims 2, 4, and 5 under 35 U.S.C. § 103(a) as being unpatentable over Stancil in view of Luke and in further view of Applicant's Admitted Prior Art (AAPA). Appellant respectfully submits that the Examiner's rejection is erroneous for at least the following reasons.

Claims 2, 4, and 5 each depend from claim 1, and thus incorporate its features. Accordingly, for at least the same reasons as given above with respect to claim 1, Appellant respectfully requests reversal of the § 103 rejection of claims 2, 4, and 5.

E. Claims 11, 13, and 14

The Examiner rejected claims 11, 13, and 14 under 35 U.S.C. § 103(a) as being unpatentable over Stancil in view of Luke and in further view of Applicant's Admitted Prior Art (AAPA). Appellant respectfully submits that the Examiner's rejection is erroneous for at least the following reasons.

Claims 11, 13, and 14 each depend from claim 10, and thus incorporate its features. Accordingly, for at least the same reasons as given above with respect to claim 10, Appellant respectfully requests reversal of the § 103 rejection of claims 11, 13, and 14.

F. Claims 20 and 22-27

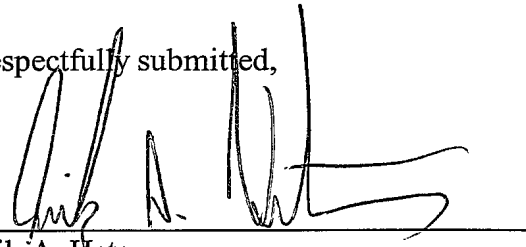
The Examiner rejected claims 20 and 22-27 under 35 U.S.C. § 103(a) as being unpatentable over Stancil in view of Luke and in further view of Applicant's Admitted Prior Art (AAPA). Appellant respectfully submits that the Examiner's rejection is erroneous for at least the following reasons.

Claims 20, and 22-27 each depend from claim 19, and thus incorporate its features. Accordingly, for at least the same reasons as given above with respect to claim 19, Appellant respectfully requests reversal of the § 103 rejection of claims 20 and 22-27.

VIII. CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejection of claims 1-2, 4-11, 13-20, 22-31 and 96 was erroneous, and reversal of his decision is respectfully requested.

Respectfully submitted,



Erik A. Heter
Reg. No. 50,652
AGENT FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8800

Date: 10/1/04

IX. APPENDIX

The claims on appeal are as follows.

1. An SMBus host controller comprising:

an SMBus interface; and

an SMBus message handler including:

a memory storing microcode comprising at least two programs each for handling a bus command protocol, each program comprising at least one instruction;

an interface to a register configured to identify a starting address of a program in said memory;

an instruction fetch unit configured to read an instruction at an address in said memory, said address being specified by a program counter;

a finite-state machine configured to receive and interpret the instructions read by said instruction fetch unit and manage the data transfer between the SMBus interface and a register set in compliance with said instructions read from said memory; and

an address register array comprising a plurality of starting addresses of programs stored in said memory, said register comprising an offset for pointing at a specific register in said address register array.
2. The SMBus host controller of claim 1, wherein said register set complies with the ACPI specification.
4. The SMBus host controller of claim 2, further comprising a buffer pointer register for pointing at one of a plurality of data registers said finite-state machine transferring data read from the SMBus interface to the data register at which said buffer pointer register points if said finite-state machine interprets a reception

instruction; said finite-state machine transferring the data read from the data register at which said buffer pointer register points to said SMBus interface if said finite-state machine interprets a transmission instruction.

5. The SMBus host controller of claim 4, wherein the finite-state machine causes said buffer pointer register to be incremented each time a reception instruction or a transmission instruction is executed.
6. The SMBus host controller of claim 1, further comprising a loop counter for storing the value of a block counter register in said loop counter if the finite-state machine executes an instruction to transmit data from said block counter register to said SMBus interface, said loop counter being decremented each time a data byte is transmitted to said SMBus interface while an instruction to transmit data from said block counter register to said SMBus interface is executed, wherein the instruction to transmit data from said block counter register to said SMBus interface is completed when the value of said loop counter reaches zero.
7. The SMBus host controller of claim 1, further comprising a loop counter and a block count register both for storing a byte received from said SMBus interface if the finite-state machine executes an instruction to receive data at said block counter register from said SMBus interface, said loop counter being decremented each time a data byte is transmitted to or received from said SMBus interface while an instruction to receive data at said block counter register from said SMBus interface is executed, wherein the instruction to receive data at said block counter register from said SMBus interface is completed when the value of said loop counter reaches zero.
8. The SMBus host controller of claim 1, wherein each instruction comprises one bit indicating as to whether or not an instruction is the last instruction in the program.
9. The SMBus host controller of claim 1, wherein each instruction comprises one bit indicating as to whether an instruction is to be executed only once or this instruction is to be executed repeatedly until a loop counter becomes zero,

wherein said loop counter is decremented each time an instruction is executed repeatedly.

10. An integrated circuit chip for transmitting and receiving data over an SMBus, the integrated circuit chip comprising:

an SMBus interface;

an interface to a memory storing microcode comprising at least two programs each for handling a bus command protocol, each program comprising at least one instruction;

an interface to a register configured to identify a starting address of a program in said memory;

an instruction fetch unit configured to read an instruction at an address in said memory; said address being specified by a program counter (pc);

a finite-state machine configured to receive and interpret the instructions read by said instruction fetch unit and to manage the data transfer between the SMBus interface, and a register set in compliance with said instructions read from said memory;

an address register array comprising a plurality of starting addresses of programs stored in said memory, said register comprising an offset for pointing at a specific register in said address register array.

11. The integrated circuit chip of claim 10, wherein said register set complies with the ACPI specification.

13. The integrated circuit chip of claim 11, further comprising a buffer pointer register for pointing at one of a plurality of data registers comprised in register set; said finite-state machine transferring data read from the SMBus interface to the data register at which said buffer pointer register points if said finite-state machine

interprets a data reception instruction, said finite-state machine transferring the data read from the data register at which said buffer pointer register points to said SMBus interface if said finite-state machine interprets a data transmission instruction.

14. The integrated circuit chip of claim 13, wherein the finite-state machine causes said buffer pointer register to be incremented each time a data reception instruction or a data transmission instruction is executed.
15. The integrated circuit chip of claim 10, further comprising a loop counter for storing the value of a block counter register SMB_BCNT in said loop counter if the finite-state machine executes an instruction to transmit data from said block counter register to said SMBus interface, said loop counter being decremented each time a data byte is transmitted to said SMBus interface while an instruction to transmit data from said block counter register to said SMBus interface is executed, wherein the instruction to transmit data from said block counter register to said SMBus interface is completed when the value said loop counter reaches zero.
16. The integrated circuit chip of claim 10, further comprising a loop counter and a block count register comprised in said register set both for storing a byte received from said SMBus interface if the finite-state machine executes an instruction to receive data at said block counter register from said SMBus interface, said loop counter being decremented each time a data byte is transmitted to or received from said SMBus interface while an instruction to ~~a~~-receive data at said block counter register is executed, wherein the instruction to receive data at said block counter register from said SMBus interface is completed when the value of said loop counter reaches zero.
17. The integrated circuit chip of claim 10, wherein each instruction comprises one bit indicating as to whether or not an instruction is the last instruction in the program.

18. The integrated circuit chip of claim 10, wherein each instruction comprises one bit indicating as to whether an instruction is to be executed only once or this instruction is to be executed repeatedly until a loop counter becomes zero, wherein said loop counter is decremented each time an instruction is executed repeatedly.
19. A method for controlling an SMBus, the method comprising:
- storing, in a memory, microcode comprising at least two programs each for handling a bus command protocol, each program comprising at least one instruction;
- identifying a starting address of one of the at least two programs stored in the memory;
- fetching instructions of said program one after another and receiving the instructions into a finite-state machine;
- interpreting the instructions;
- transferring data between an SMBus interface and a register set in compliance with the instruction present in said finite-state machine;
- reading a first value of a protocol register specifying an offset in an address register array; and
- reading a second value of a register of said address register array, said register being specified by said offset; said second value constituting said starting address of said program.
20. The method of claim 19, wherein said register set complies with the ACPI specification.

22. The method of claim 20, wherein said transferring step comprising the sub-steps of:
- interpreting a reception instruction;
- reading the value of a buffer pointer register; and
- transferring the data read from said SMBus interface to the data register at which the value stored in said buffer pointer register points.
23. The method of claim 22, wherein said transferring step further comprises incrementing said value of said buffer pointer register.
24. The method of claim 22, wherein said transferring step further comprising decrementing a loop counter and checking as to whether said loop counter has a value of zero.
25. The method of claim 20, wherein said transferring step comprises the sub-steps of:
- interpreting a transmission instruction;
- reading the value of a buffer pointer register; and
- transferring the data read from the data register at which the value stored in said buffer pointer register to said SMBus interface.
26. The method of claim 25, wherein said transferring step further comprises incrementing of said buffer pointer register.
27. The method of claim 25, wherein the transferring step further comprises decrementing of said loop counter.
28. The method of claim 19, wherein said transferring step comprises:

Interpreting an instruction to transmit data from a block counter register to said SMBus interface;

storing the value of said block count register in a loop counter; and

transmitting the value of said block counter register to said SMBus interface.

29. The method of claim 19, wherein said transferring step comprises:

interpreting an instruction to receive data from said SMBus interface;

transmitting a byte from said SMBus interface to said block count register; and

storing the value of the byte received from said SMBus interface to a loop counter register.

30. The method of claim 19, wherein said transferring further comprises:

determining as to whether a stop bit has a predetermined value; if this is the case:

writing 80h into a status register of said register set.

31. The method of claim 19, wherein said transferring further comprises:

determining as to whether a loop bit of an instruction has a predetermined value;
if that is the case,

executing said instruction repeatedly;

decrementing a loop counter each time said instruction is executed;

finishing the execution of said loop instruction when the value of the said loop counter becomes zero; and

fetching the next instruction.

96. The SMBus host controller of claim 1, wherein the memory storing microcode is a read-only memory.

X. EVIDENCE APPENDIX

No evidence submitted under 37 C.F.R. §§ 1.130, 1.131, or 1.132 or otherwise entered by the Examiner is relied upon in this appeal.

XI. RELATED PROCEEDINGS APPENDIX

There are no related proceedings.